Energy Efficient Path Planning for Unmanned Surface Vehicle in Spatially-Temporally Variant Environment

Hanlin Niu[a,b], Ze Ji[a,*], Al Savvaris[b], Antonios Tsourdos[b]

[a]*School of Engineering, Cardiff University, Cardiff CF24 3AA, United Kingdom*
[b]*School of Aerospace, Transport and Manufacturing, Cranfield University, Bedford, Bedfordshire, MK43 0AL, United Kingdom*

Abstract

Unmanned Surface Vehicles (USVs) are increasingly used for ocean missions, which typically require long duration of operations under strict energy constraints. Consequently, there is an increased interest in energy efficient path planning for USVs. This work proposes a novel energy efficient path planning algorithm to address the challenges with the presence of spatially-temporally variant sea current and complex geographic map data, by integrating the following algorithms, namely Voronoi roadmap, Dijkstras searching, coastline expanding and genetic algorithm. The selection, crossover and mutation operators are employed as part of the GA algorithm. The dividing, smoothing and exchanging operators are proposed to improve the quality of the path and adapt to the Voronoi-Visibility roadmap. The Global Self-Consistent Hierarchical High-Resolution Shorelines dataset and historical sea current dataset are applied to demonstrate the flexibility and practicability of the proposed algorithm. To evaluate the performance, the Voronoi-GA energy efficient algorithm and Voronoi-Visibility energy efficient path re-planning algorithm are also implemented to provide the baseline for comparison. The proposed algorithm generates the most energy efficient paths in ten USV missions, while keeping a configurable clearance from the coastlines. The practicability and scalability

*Corresponding author
Email addresses: NiuH1@cardiff.ac.uk (Hanlin Niu), JiZ1@cardiff.ac.uk (Ze Ji), a.savvaris@cranfield.ac.uk (Al Savvaris), a.tsourdos@cranfield.ac.uk (Antonios Tsourdos)

of this algorithm is also demonstrated by analysing the computational time in these ten missions.
Keywords: Unmanned Surface Vehicle; Voronoi Diagram; Visibility Graph; Dijkstra's Search Algorithm; Genetic Algorithm; Energy Efficient; Path Planning; Spatially-Temporally Variant; Sea Current.

1. Introduction

Energy efficiency is an important characteristic of path planning algorithms for autonomous systems. In particular, deployment of Unmanned Surface Vehicle (USV) (Lv et al., 2019) and Unmanned Underwater Vehicle (UUV) (Palomeras et al., 2018; Xia et al., 2019) for missions with various environmental disturbances and uncertainties in presence make path planning more challenging, because of the existence of uncertain, inaccurate and dynamic environmental information. The path planning algorithms of USVs and UUVs aim to optimise the following aspects: travel time, energy consumption and safety. Typically, USVs and UUVs are deployed for long term missions with limited energy. As a result, there is an increasing interest in developing energy efficient path planning algorithm in recent years (Lee et al., 2015; Zeng et al., 2015). In this research, we focus on optimising the energy consumption and safety of the USV.

A level-set based approach for UUV path planning, where the external flow was explicitly accounted for was described in (Lolla et al., 2012, 2014, 2015). In the level-set approach, time-optimal trajectories for single or multiple vessels are determined by employing a level-set expansion of the flow field given the desired start and goal positions for the vehicles. The underlying motion model behind the level-set method includes a velocity model, which is very often restrictive. Since the strategy requires the entire information of the flow field and requires massive amounts of computational power, when performing the various level set expansions, the strategy will not be amenable for the real time planning purpose (Kularatne et al., 2016; Singh et al., 2018).

Heuristic grid searching approaches are commonly applied in solving NP-

hard and multi-objective optimisation problems. The operation space is transformed into a graph representation using quad trees in (Carroll et al., 1992), so that optimal paths can be produced using the A* algorithm. An A* based energy efficient path planning algorithm (Garau et al., 2014) was proposed to consider sea current data, in which situation the thrust was assumed to be constant. Another A* based path planning algorithm of AUVs was applied in (Garau et al., 2005) to generate energy optimal paths in conditions of different eddy currents, and this work was tested in a simulated ocean environment. The effect of different heuristic functions on the results was analysed. However, deterministic and heuristic methods like the A* algorithm (Le et al., 2018) (Song et al., 2019) requires massive computational power and are criticised for their weakness in dealing with high-dimensional problems (Zadeh et al., 2016).

Dynamic programming (Bryson and Ho, 1975) has been used as a graph-based searching method, where a cost is related to the edge of a graph. Although dynamic programming is able to generate optimal paths, the computational cost will increase geometrically with the dimension of the solution space. A mixed integer linear programming (MILP) path planning algorithm was introduced to navigate multiple AUVs (Yilmaz et al., 2008). However, the computational time of this approach will increase exponentially with the problem size, and thus this approach will be limited for real-world applications (Eichhorn, 2015). In the case of USV path planning in a spatially variant and temporally variant sea current environment, dynamic programming and MILP may not be computationally feasible.

Path planning algorithm based on a searching strategy from Darwin's theories of evolution was introduced in (Goldberg and Holland, 1988). The Genetic Algorithm (GA) (Kim et al., 2017; Chen et al., 2018; Fu et al., 2018) and its inheritance are the most popular examples in this area, where a specific number of candidate paths are maintained and these paths are iteratively evolved and selected by applying genetic operators. GAs have also been applied in the AUV path planning problem (Alvarez et al., 2004; Zadeh et al., 2016). The computational complexity of GAs increases linearly with the dimension of the solution

space, which is more computationally efficient than dynamic programming algorithm (Alvarez et al., 2004), hence more advantageous for searching in high dimensional dataset. Previously proposed GAs (Alvarez et al., 2004; Zadeh et al., 2016) applied grid maps or cell maps to represent collision free space and applied the random walk algorithm to generate the initial population. However, in USV path planning scenarios, the mission operations could take place in quite complex spatial environments. For instance, there are about 98 islands in Singapore. In such complex spatial environments, the random walk method is inefficient in finding feasible paths for given start locations and destinations (Zhang et al., 2008). In the work of Song et al. (2017), a multi-layered fast marching (MFM) algorithm was proposed for path re-planning to optimise the energy efficiency at different time steps using updated sea current data, where the spatial environment is represented by a grid map. A global USV path planning algorithm based on an improved ant colony algorithm was proposed by Song (2014) to compute a collision-free and smooth path. Although this algorithm is computationally efficient, it does not take into account the sea current and it also uses grid map for representing a simple spatial environment. Same as the GA algorithm, particle swarm optimisation (PSO) algorithm is also subordinate to the heuristic approach. A dynamic augmented multi-objective PSO (MOPSO) (Ma et al., 2018) was proposed to generate energy efficient path taking into account multiple objectives and constraints. However, the computational efficiency of the algorithm was not analysed. Tang et al. (2016) integrated PSO with the differential evolution (DE) algorithm to solve the global path planning problem efficiently. The algorithm was compared with four state-of-art evolutionary algorithms, including JADE (Yang et al., 2015), TVPSO(Khalili-Damghani et al., 2013), GS (Li and Duan, 2012) and mGA (Shiltagh and Jalal, 2013), and it was validated that the new algorithm outperforms the other four algorithms in terms of path optimality. However, the energy consumption was not taken into account. Moreover, the work of Ma et al. (2018) and Tang et al. (2016) just uses grid map and a few simple polygons to represent a simple environment and obstacles respectively, and the flexibility and computational efficiency of

the algorithm dealing with complex environment were not analysed.

Prior work related to energy efficient path planning algorithms mainly employs grid maps to represent spatial environment due to its simplicity in implementation that discretises the space into independent cells. However, in complex spatial environments, such as Singapore Strait that has 98 islands, it is difficult to apply grid maps with a fixed predefined grid size to correctly represent the environmental occupancy state. The approximate cell decomposition approach, which is also known as the quadtree decomposition, can be applied to solve this problem by subdividing the mixed obstacle and free regions into four quarters iteratively until a certain level of resolution is achieved. However, this approach is not computationally efficient in the case of large-scale spatial environments where islands' sizes and their inter-distances are highly in-homogeneous. Compared to the grid occupancy map, the Voronoi roadmap is known to be more computationally efficient for map representation in the two-dimensional space (Pehlivanoglu, 2012) and better in complex spatial environments (Benavides et al., 2011). The Voronoi-Visibility roadmap and Dijkstra's algorithm were integrated for generating the energy efficient path in (Niu et al., 2018). The efficiency of the algorithm in terms of its computational demand and energy consumption was evaluated by comparing with the Voronoi-Visibility shortest path method. However, this algorithm has only taken spatially variant sea currents into consideration and has not addressed the temporally variant sea current problem. Benavides et al. (2011) introduced a path planning algorithm by combining the Voronoi roadmap and GA method, but without taking into account energy consumption.

In this work, we combine the ideas of Benavides et al. (2011) and Niu et al. (2018), and propose the Voronoi-Visibility-GA energy efficient (VVGAEE) path planning algorithm. The motivation for the combination in this research is to keep the flexibility and computational efficiency of the Voronoi diagram algorithm in generating the collision free roadmap and also the computational efficiency of the GA algorithm in searching in high dimensional dataset, so as to generate energy efficient paths in spatially-temporally variant environments.

Inspired by the work of Benavides et al. (2011) and Song et al. (2017), two other methods, named Voronoi-GA energy efficient (VGAEE) path planning algorithm and Voronoi-Visibility energy efficient path re-planning (VVEEPRE) algorithm, are also proposed and developed for performance comparison. Finally, the Voronoi-Visibility energy efficient (VVEE) algorithm (Niu et al., 2018), VGAEE, VVEEPRE and VVGAEE algorithms are compared using ten USV missions in a complex geographic environment (98 islands) and spatially-temporally variant sea current scenarios.

The remainder of this paper has the following structure: Section 2 describes the problem statement. The methodology is presented in Section 3, which includes the Voronoi roadmap generation and GA algorithm implementation. The proposed algorithm is compared with three algorithms in ten USV missions in Section 4. The conclusion and the future work are presented in Section 5.

2. Problem statement

2.1. Large spatial dataset

A high-resolution navigation map is obviously beneficial to the performance in path planning accuracy, but would also introduce high computational burden. For instance, as shown in Fig. 1, there are 98 islands in Singapore, which are represented by 4128 vertices. Using the Visibility graph algorithm as an example, the computational cost will be $O(n^2)$ time and the edge number generated will be $O(n^2)$ at its worst, where n represents the number of vertices. Processing such a large number of edges would require a massive amount of computational time. Choosing a computationally efficient roadmap generation algorithm is needed in practice.

2.2. Spatially-Temporally variant sea current

In long term USV missions, the sea current will vary with both location and time. As an example, the sea current state changes of Singapore strait from 0:00 am to 05:00 am on the 11th June of 2014 are shown in Fig. 2.

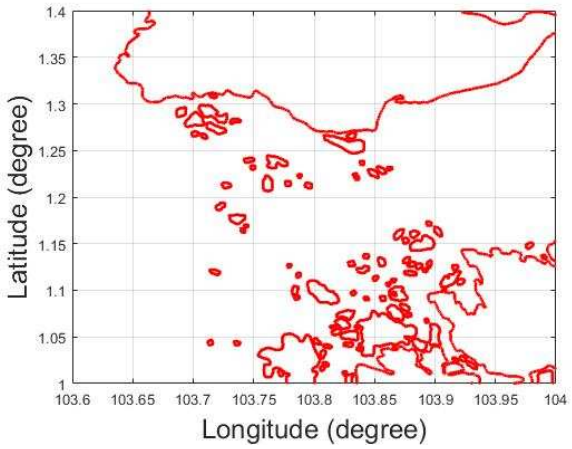
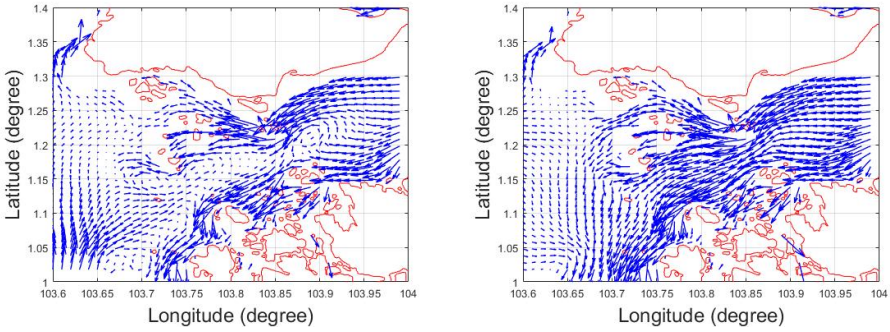


Fig. 1. Singapore islands

From Fig. 2a to Fig. 2f, it can be seen that the tide begins to rise at 0:00 am and the water came into Singapore strait from the East. At 2:00 am, more and more water came into the west part of Singapore strait. The tide on the west side began to rise. Comparing Fig. 2a and Fig. 2f, we can see the large difference between 0:00 am and 5:00 am. If the USV encounters dramatically temporally variant sea current state, it will be necessary for an USV path planning algorithm to consider not only the spatial variance of the sea current but also the temporal variance, which will save the total energy consumption during the whole USV mission.

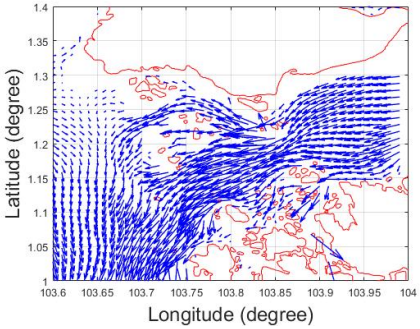
2.3. Clearance distance c

It is necessary to keep the USV a clearance distance c away from the coastlines to maximise its safety in autonomous navigation, because areas close to the coastlines are more likely to have crowded traffics, and also more hazadous due to the higher probability of shallow water. Also, the spatial accuracy of a map is dependent on the techniques used in the mapping processes that will introduce various levels of uncertainty. Such inaccuracy with the map data will need to be taken into account. For instance, Singapore islands plotted in Fig. 3 are

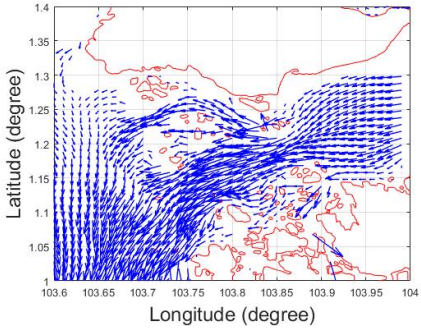


(a) Sea current state at 00:00am

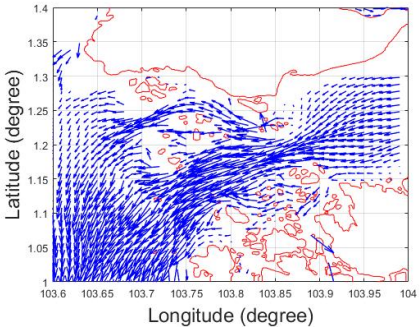
(b) Sea current state at 01:00am



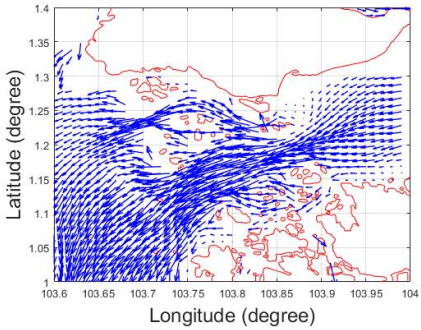
(c) Sea current state at 02:00am



(d) Sea current state at 03:00am



(e) Sea current state at 04:00am



(f) Sea current state at 05:00am

Fig. 2. Illustration of the Singapore sea current state on the 11th June 2014: The sea current state is depicted with the blue arrows and the island coastlines are outlined in red.

using the GSHHS data. However, mapping the island profile directly into the Google® map, as shown in Fig. 4, we can clearly see the spatial offset between two profiles of the same island.

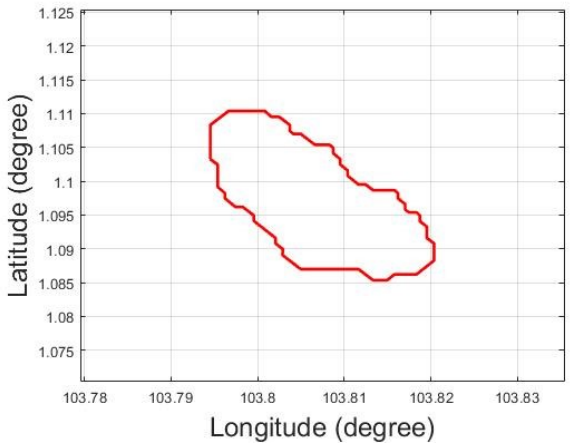


Fig. 3. Singapore island

Fig. 4. Illustration of inaccuracy with the spatial data: The green line outlines the island profile using the GSHHS dataset and the underlayer map represents the corresponding part on the Google® map.

3. Methodology

In this work, a novel Voronoi roadmap and GA based energy efficient path planning algorithm is proposed for situations with spatially-temporally variant sea current. The algorithm includes two main steps: 1) collision free roadmap generation, and 2) GA based searching for collision free and energy efficient path. This section is organised as follow. The algorithm architecture is presented in section 3.1. Section 3.2 introduces the environmental dataset used in this work. The Voronoi-based collision free roadmap generation is presented in section 3.3, followed by the GA implementation in section 3.4.

3.1. Algorithm Architecture

Voronoi roadmaps are generated in $O(n)$ time, which is more computationally efficient than the Visibility graph ($O(n^2)$ time). The Voronoi-based roadmap is firstly generated, then the island coastlines are expanded by applying a coastline expanding algorithm, which will keep the USV away from nearby islands with a user-configurable distance. The Voronoi diagram is applied to generate roadmap around the islands. However, not all paths of the roadmap are reachable. Such unreachable paths are then deleted accordingly.

After the generation of the collision-free roadmap, the Dijkstra's and Visibility graph algorithms are used to generate the first generation of chromosomes for the GA. The GA is implemented with the objective of finding a collision free and energy efficient path with the given start point and the destination. A fitness function is introduced for above optimisation criterion and the fitness function will be applied for evaluating all the individuals of each generation. In the GA implementation, the selection, crossover, and mutation operators are applied. By using the selection operator, the individuals those have the best fitness value will be selected and stored. The crossover operator shuffles the existing population solutions and generate more possible candidates. And the mutation operator is used for increasing the diversity of the population. The evaluation, selection, crossover and mutation are applied in a loop. Until the

termination condition is satisfied, the iteration will stop and the best individual will be selected.

3.2. Environmental dataset

The environmental dataset applied in this work includes the island coastlines dataset and the sea current dataset.

3.2.1. Coastline data

The Global-Consistent Hierachical High-Resolution Shorelines (GSHHS) is a publicly available data set, providing up-to-date shoreline data in 5 resolutions: full-resolution (f), high-resolution (h), intermediate-resolution (i), low-resolution (l) and crude-resolution (c). In this work, we use full-resolution (f), the highest available resolution at 100 metres, to maximise the details to be used in the work. High resolution data can provide a more realistic geographic environment for evaluating the flexibility and practicability of the proposed method in addressing different challenges in complex geographic scenarios.

3.2.2. Sea currents

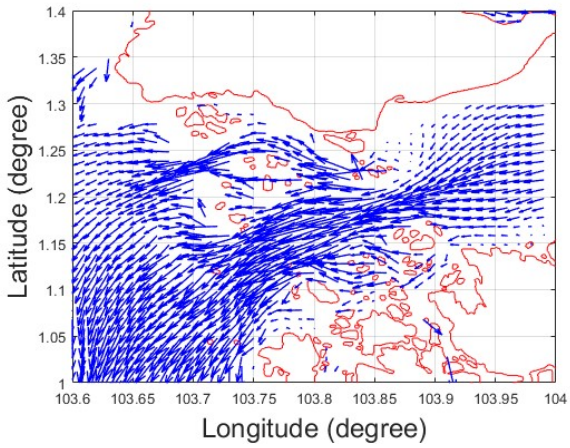


Fig. 5. Sea current state of Singapore Strait: The current state was recorded at 5:00am, June 11, 2014. The island coastlines are outlined by the red lines and the blue arrows depict the sea current state

Table 1: Specifications of the sea current data

Region	Parameters	Resolution	Updating Time Step	Forecast Length
North Atlantic	Current	11 km	24 hrs	144 hrs
Gulf Stream	Current	11 km	24 hrs	144 hrs
English Channel	Current	2 km	15 min	48 hrs
North West Europe	Current	20 km	60 min	120 hrs
Singapore Strait	Tide Current	1.1 km	60 min	48 hrs

The sea current data were obtained from the company TideTech Ltd TideTech (2017). Table 1 shows specifications of the sea current data in terms of resolutions, time steps and forecast lengths. In this work, Singapore Strait was used as the USV mission area, for the reason that there are 98 islands that is complex enough for testing the flexibility and practicability of the proposed algorithm. Fig. 5 shows sea current and coastline data of Singapore Strait.

3.3. Voronoi collision free roadmap generation

The Voronoi roadmap for generating collision free roadmaps includes the following steps: coastline expanding algorithm, Voronoi diagram algorithm and unreachable paths removal. The collision free roadmap generation with a predefined clearance c is explained in details in (Niu et al., 2018), that the coastlines of the islands should be translated and expanded with a certain distance. Such generated candidate paths should not intersect with the expanded coastlines. An example result of expanded coastlines is shown in Fig. 6.

After the coastlines are expanded, the Voronoi diagram is created, as shown in Fig. 7. We can see that the density of the Voronoi edges is very high in many areas, because of the high resolution data we used. However, not all the edges are available and some of them intersect with the islands, which will be unreachable for the USV. In this work, the edges will be removed if the following two conditions are met: 1) if the path segment intersects with any of the expanded coastlines; and 2) if the path is connected to a Voronoi node that is inside the coastline profile. A clearer roadmap will be constructed after the unreachable edges are removed, as shown in Fig. 8.

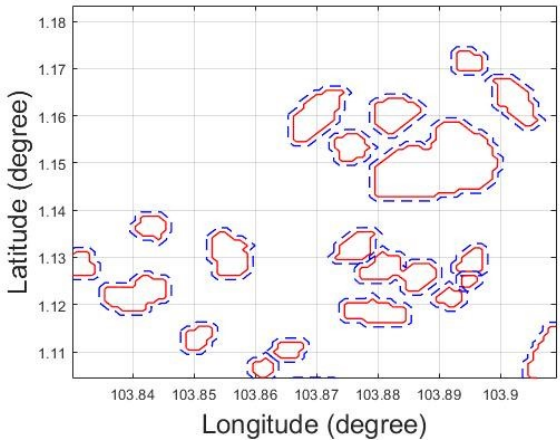


Fig. 6. Illustration of coastline expanding algorithm implementation in Singapore Strait: The red lines represent the original coastlines and the blue dash lines represent the expanded coastlines.

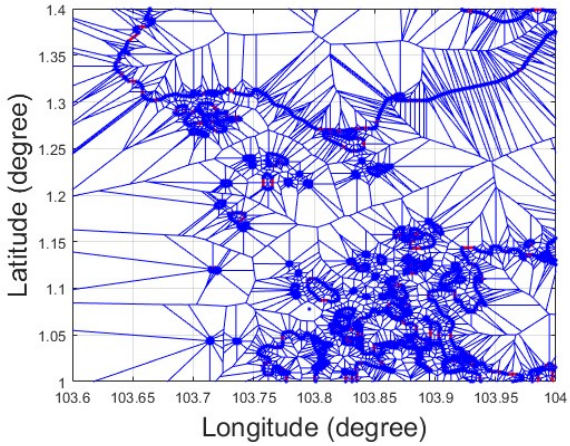


Fig. 7. Voronoi roadmap of the Singapore islands: The red lines represent the expanded coastlines and the blue line are the Voronoi edges

3.4. GA implementation

3.4.1. Initialization

The first generation feasible paths are produced using the VVEE algorithm (Niu et al., 2018), as summarised below. After the Voronoi collision

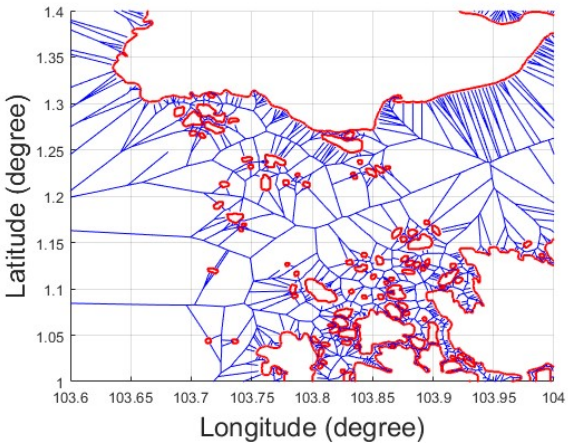


Fig. 8. Voronoi collision free roadmap of the Singapore islands: The red lines represent the expanded coastlines and the blue lines are the collision free Voronoi edges

free roadmap is generated, an energy cost function will be applied to estimate the energy consumption weights of all Voronoi edges. Then the energy efficient Voronoi path will be provided by direct application of the Dijkstra's algorithm. However, as the Voronoi graph contains many redundant waypoints, the Visibility graph is applied to optimise the path. Finally, an energy efficient path is produced by applying the Dijkstra's algorithm again. The VVEE algorithm was designed for environment with temporally-invariant sea current. Therefore, we estimate the mission duration and apply the sea current data of each time updating interval to produce paths of the first generation. Note that the mission duration can be estimated by calculating the length of the first path and calculating the travelling duration of the whole course. The mission duration is not essential to be perfectly accurate. For instance, with Singapore strait, the sea current data will be updated each hour, if the USV needs T hours for travelling, the sea current data of every hour can be used for generating T feasible paths. These T feasible paths will be used for generating N feasible paths as the first generation by applying the mutation operator and the crossover operator.

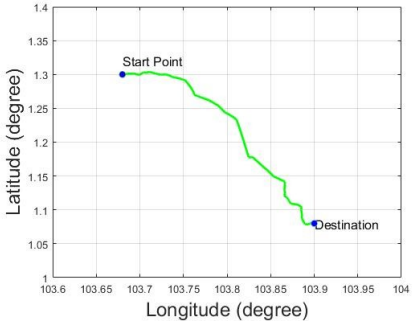
3.4.2. Evaluation

The quality of a path is evaluated by the fitness function, which measures the energy cost of each path in the population. There are two steps of calculating the energy cost in the spatially-temporally variant sea current. Firstly, we need to divide the path into equidistant segments based on the USV travelling distance of one hour. Assume the USV is commanded to travel from $(103.68, 1.3)$ to $(103.9, 1.08)$ at 0:00am on 11/06/2014 in Singapore strait. We have a candidate path, as shown in Fig. 9a. The update interval of the sea current data is one hour TideTech (2017). Since the USV is commanded to travel at 1 m/s, we first decompose the whole path into several equidistant segments, where the length of each segment is determined by the travelling distance within one hour. As shown in Fig. 9b, the end points of the path segments are denoted by the red points, called one hour break points. The second step is to calculate the energy cost of each equidistant path. When the USV is travelling on each equidistant path, we assume the sea current keeps constant in the corresponding time interval. For example, when the USV is travelling within the one hour break points 3 and 4, we will use the sea current data of 3:00am to calculate the corresponding energy cost. The illustrations are given in Fig. 9c and Fig. 9d. In Fig. 9c, the path from point 3 to point 4 is illustrated in the red rectangle area and the corresponding sea current data at 3:00am is required. Fig. 9d illustrates the sea current state.

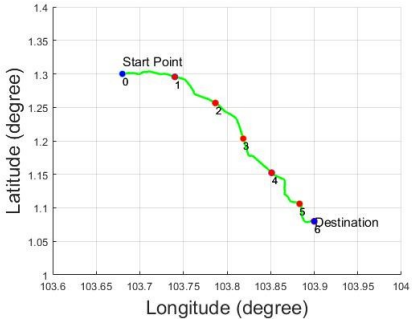
The method to calculate the energy cost of the equidistant path is given below. Assume an USV is operating on one equidistance path segment, denoted by N_iN_{i+1}. The USV ground speed $\overrightarrow{v_g}$, the sea current speed $\overrightarrow{v_c}$, and the relative USV speed $\overrightarrow{v_u}$ satisfy Eq. (1).

$$\overrightarrow{v_g} = \overrightarrow{v_u} + \overrightarrow{v_c} \tag{1}$$

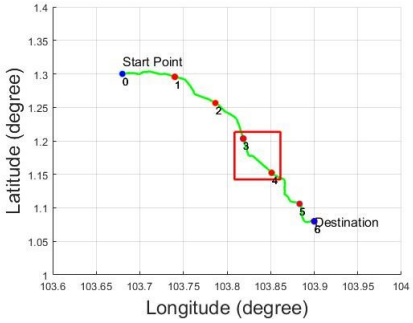
In this research, the USV is assumed to travel at a constant ground velocity $|v_g|$. Provided the direction of $\overrightarrow{v_g}$ and sea current $\overrightarrow{v_c}$ are known, $\overrightarrow{v_u}$ can be calculated using Eq. (1). The hydrodynamic drag F_d can be calculated by Eq.



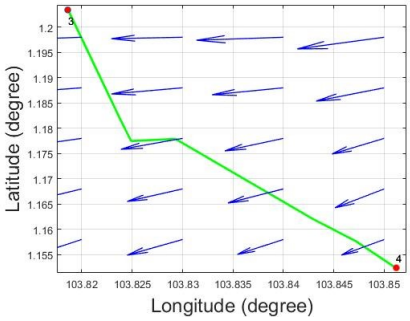
(a) Candidate path



(b) Candidate path with one hour break point



(c) Candidate path with highlight area



(d) Sea current state from point 3 to point 4

Fig. 9. Illustration of evaluation

(2).

$$F_d = \frac{1}{2}\rho|v_u|^2 C_D A \tag{2}$$

where ρ denotes the water density, C_D and A represent drag coefficient and reference area, respectively. Eq. (3) is used to estimate the USV energy consumption weight E:

$$E = F_d \times |v_u| \times t \tag{3}$$

The travel duration t specifies the time needed for the USV to travel between two adjacent waypoints, N_i and N_{i+1} (see Eq. (4)):

$$t = \frac{\left|\overline{N_i N_{i+1}}\right|}{v_g} \tag{4}$$

Then we can derive Eq. (5):

$$E = \alpha|v_u|^3 \frac{\left|\overline{N_i N_{i+1}}\right|}{v_g} \tag{5}$$

where α is the combination of three values: ρ, C_D, and A. Considering α is constant, we can simplify the calculation by assuming α to be 1 here. Therefore, in the case that $\overrightarrow{v_g}$ is a constant value, there remained only two values that need to be calculated: $\overrightarrow{v_u}$ and $\left|\overline{N_i N_{i+1}}\right|$. The path of $N_i N_{i+1}$ is divided into equidistant segment based on the USV travelling distance in each hour. The total energy consumption can be estimated by using the updated sea current data in each hour. In case that the path intersects with any of the expanded coastlines, the fitness value of this path will be set to a very large number.

3.4.3. Crossover

The next step is to evaluate the performance of N feasible paths by calculating and ranking their fitness values. The top 40% of the ranked paths will be selected as parents for generating new individuals by applying the crossover operator. The crossover operator exchanges the genetic information of the parents. Each path consists of a sequence of waypoints, which will be indexed.

The indices of the waypoints are used to represent the gene of the corresponding path. Assuming we have two parents p_{w1} and p_{w2} randomly selected, the waypoint indices of these two parents can be denoted by Eq. (6) and Eq. (7).

$$p_{w1} = \left\{ p_{w1,1}, ..., p_{w1,i}, ..., p_{w1,m} \right\} \tag{6}$$

$$p_{w2} = \left\{ p_{w2,1}, ..., p_{w2,j}, ..., p_{w2,n} \right\} \tag{7}$$

where $p_{w1,i}$ denotes the i_{th} waypoint of parent p_{w1}. We randomly choose two waypoints from the two parents, indexed as i and j, respectively, where $1 \leq i \leq m$ and $1 \leq j \leq n$. Each parent will be split into two subsets of waypoints at the selected index correspondingly. Then a new pair of offspring can be generated by swapping the second subsets of p_{w1} and p_{w2}, as shown in Eq. (8) and Eq. (9). The crossover operator shuffles the existing populations and searches for a better solution space.

$$p_{wo1} = \left\{ \left[p_{w1,1}, ..., p_{w1,i} \right], \left[p_{w2,j+1}, ..., p_{w2,n} \right] \right\} \tag{8}$$

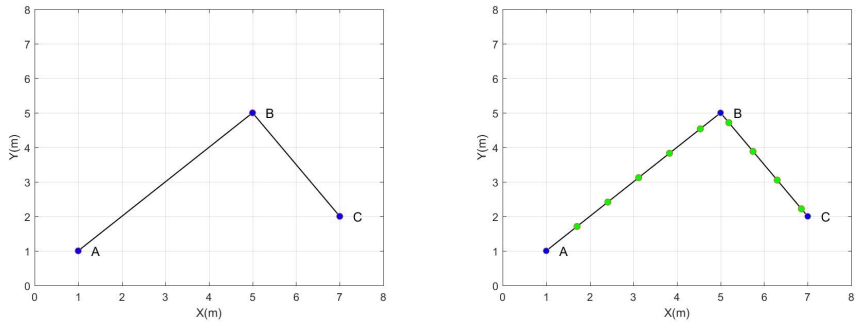
$$p_{wo2} = \left\{ \left[p_{w2,1}, ..., p_{w2,j} \right], \left[p_{w1,i+1}, ..., p_{w1,m} \right] \right\} \tag{9}$$

3.4.4. Mutation

Mutation is usually used to increase the population diversity and avoid the solutions to converge prematurely to a stable local minimum (Alvarez et al., 2004). Three mutation operators are used in this work: dividing, smoothing and exchanging.

(a) The dividing operator

Firstly, if a candidate path is generated, each line segment of this path will be enquired. If the line segment is longer than a specified value L, this path will be divided to several equidistant line segments, with the equidistance equal to L, by inserting dividing waypoints, as shown in Fig. 10a and



(a) The path before applying dividing operator

(b) The path after applying dividing operator

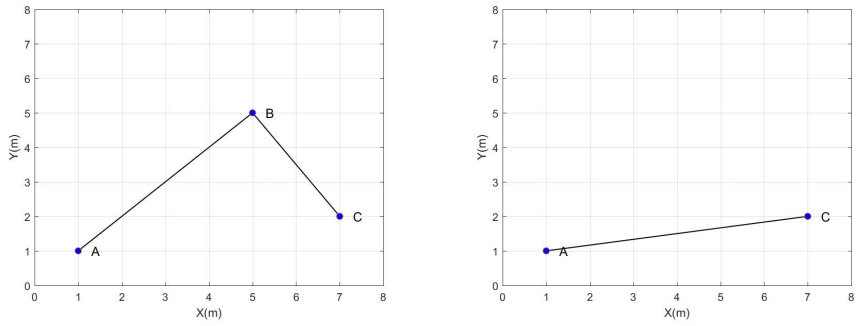
Fig. 10. Illustration of dividing operator: The blue dots represent the original waypoints and the green dots represent the new inserted waypoints.

Fig. 10b. Fig. 10a shows two connected line segments, which are named as AB and BC. We define L as the distance of 10 minutes travelling, which is set to be 1 here for demonstration. We then divide both AB and BC using the equidistant points (depicted by the green dots). The inserted waypoints are also indexed and they will be used as the new genes of the candidate path. The combination of the dividing operator and the crossover operator makes it possible for the USV to change direction in short time intervals. This operator will be applied on all the existing and newly generated candidate paths.

(b) The smoothing operator

The smoothing operator randomly removes a waypoint from a candidate path, as shown in Fig. 11a and Fig. 11b. In Fig. 11a, point B is randomly chosen from the candidate path. Since there are two points connected to it, namely A and C, after we remove B, we will directly connect A and C, creating a new merged path AC, as shown in Fig. 11b. The smoothing operator can remove redundant waypoints from the candidate paths, and can be used for generating smooth and energy efficient paths.

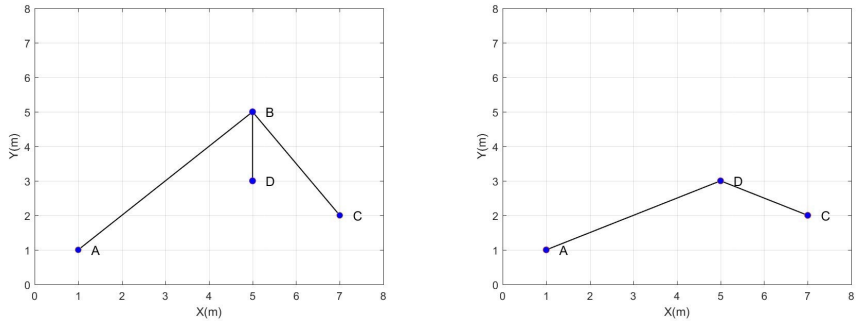
(c) The exchanging operator



(a) The path before applying smoothing operator

(b) The path after applying smoothing operator

Fig. 11. Illustration of smoothing operator



(a) The path before using exchanging operator

(b) The path after applying exchanging operator

Fig. 12. Illustration of exchanging operator

The exchanging operator will randomly exchange one waypoint to its neighbour waypoint it is connected to. Assume we have two connected line segments AB and BC, and a waypoint D is connected to B, as shown in Fig. 12a. After we applied the exchanging operator to swap B and D, our new path will be ADC, as shown in Fig. 12b. The smoothing operator or the exchanging operator will be used on half of the selected paths randomly.

3.4.5. Selection and iteration

The initial T paths are first used to generate N new paths through the processes of crossover and mutation. These paths are then evaluated by using the fitness function. A feasible energy efficient path will have a low fitness value and an inefficient path will have a high fitness value. The paths will be ranked with respect to the fitness values in ascending order. The first 40% of the ranked paths, which have the lowest fitness values, will be selected to generate $40\%N$ new individuals by applying crossover, where N is the total number of paths. Half of the selected paths, that are 20% of the total paths, will be processed through the smoothing operator or exchanging operator randomly. In each iteration, the first 40% of the paths are selected and these paths are stored, so that the most optimal paths with the best genes are kept. Together with $40\%N$ new individuals, and $20\%N$ mutation paths, we have in total N number of paths in the end of the iteration. The iteration will stop until the termination condition is met. The termination condition is simply the number of iterations, which is set as 20 in our work. Note that the parameters used in GA for crossover and mutation are configured empirically after testing the missions with different travel time. Increasing the proportion of crossover and mutated population would not necessarily further improve the performance of the algorithm and will consume more computational time. On the other hand, using a small proportion will not generate enough new paths for the selecting system.

4. Numerical Simulation and Comparison

4.1. Simulation environment

Because the sea current data of Singapore Strait can be forecast up to 48 hours in advance according to TideTech (2017), the sea current data is assumed to be known before the USV starts the mission in this research. To demonstrate the capability of the proposed algorithm in dealing with a fast changing sea current environment, simulated data is composed of some discontinuous historical

Table 2: The corresponding historical data used for simulation in the work

Simulated time interval	The time of the corresponding historical data
t_1	9:00am on 11/06/2014
t_2	12:00 (noon) on 11/06/2014
t_3	5:00pm on 11/06/2014
t_4	6:00pm on 11/06/2014
t_5	7:00pm on 11/06/2014

sea current data of Singapore Strait. The corresponding historical data sets used in our simulation are listed in Table 2.

As shown in Table 2, there are 5 time intervals simulated, and each interval is for 1 hour. The first hour of the simulation (labelled as t_1) uses the historical data of 9:00am on 11/06/2014. The second hour (t_2) uses the historical data of 12:00 pm on 11/06/2014. The rest of the simulation data can be seen in the table correspondingly. We assume the sea current remains relatively constant in each hour interval in this work. Note that this assumption is for ease of analysis and is not a restriction of the approach. The proposed algorithm is supposed to be generic and universal, regardless of the update frequency of the data.

4.2. Voronoi-Visibility energy efficient path re-planning algorithm (VVEEPRE)

In the work of Song et al. (2017), a multi-layered fast marching (MFM) path planning algorithm was proposed for avoiding collision risks and improving energy efficiency. To solve the problem of temporally variant sea current, a path re-planning strategy was applied to generate energy efficient path at each time step using the updated sea current data. In this work, the path re-planning strategy is also applied. The path planning algorithm implemented at each time step is Voronoi-Visibility energy efficient (VVEE) algorithm, which was only used for spatially variant sea state in (Niu et al., 2018). The VVEE algorithm is firstly used to generate an initial path using the sea current data of the first hour and the USV will travel until the one hour break point and re-plan the second path using the sea current data of the second time step. Following the second path, the USV will arrive at the second one hour break point. The path will be re-planned iteratively until the travelling distance between the USV and

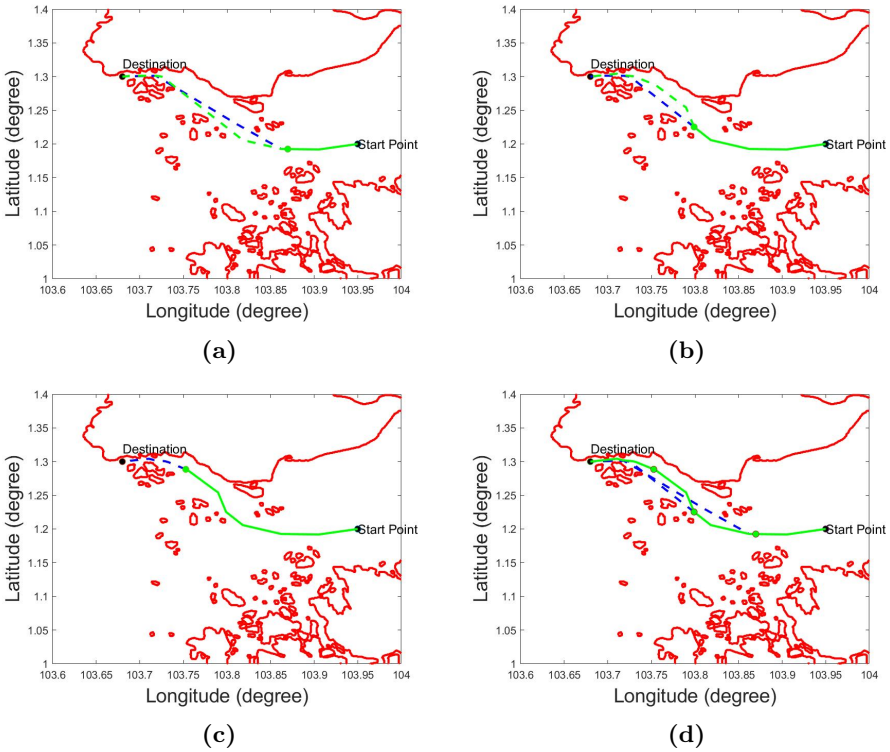


Fig. 13. Illustration of VVEEPRE algorithm: (a)-(d) show the generated paths at each time step. The blue dashed line represents the generated path using the data of last time step. The tracked trajectory is represented by the green solid line. The green dot represents the one hour break point. The green dashed line is the generated path from the one hour break point to the destination using the data of next time step. (d) shows the final tracked trajectory and all the paths of each time step

the destination is less than one hour. The new algorithm is named as Voronoi-Visibility energy efficient path re-planning algorithm (VVEEPRE).

Assume the USV is executing a mission travelling from location coordinate of $(103.95, 1.2)$ to $(103.68, 1.3)$ and the travelling speed is kept as $2.5m/s$. The results are shown in Fig. 13.

4.3. Voronoi-GA energy efficient path planning algorithm (VGAEE)

Inspired by the Voronoi-GA shortest path planning algorithm in (Benavides et al., 2011), the Vornoi-GA energy efficient (VGAEE) path planning algorithm is also developed for comparison. The VGAEE can also be treated as the mutant

algorithm of VVGAEE. The difference between VGAEE and VVGAEE is the initial energy efficient paths of VGAEE algorithm are generated by using the Voronoi roadmap instead of the Voronoi-Visibility roadmap. The parameters of the GA algorithm in VGAEE are the same as those configured in VVGAEE in Section 3. The comparison between VGAEE and VVGAEE will be used to analyse the effect of the initial paths on the final results.

Fig. 14 shows the result of the VGAEE algorithm executing the same mission as the one in section 4.2. Compared to the path generated by VVEE algorithm in Fig. 13, the initial paths (black and blue lines) generated in Voronoi roadmap have redundant waypoints. However, the final green path becomes smooth with the implementation of the proposed dividing, smoothing and exchanging operators.

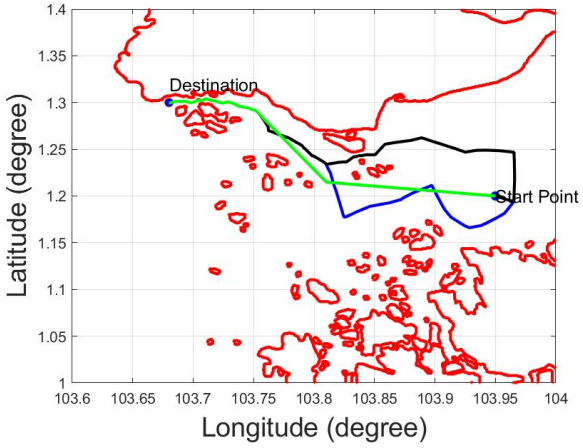
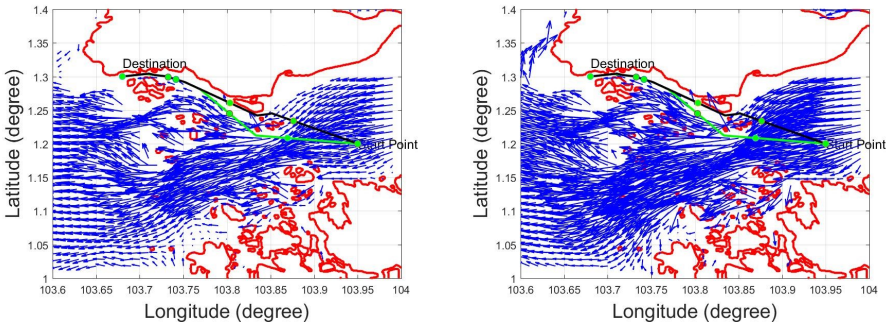


Fig. 14. VGAEE path and initial paths: The VGAEE path is represented by the green solid line. The four initial paths are represented by blue solid line, blue dashed line, black solid line and black dashed line, respectively. The first path and the second path overlap. The third path and the fourth path overlap.

4.4. Energy efficiency and computational efficiency comparison of VVEE, VVEEPRE, VGAEE and VVGAEE

Performance of the VVEE, VVEEPRE, VGAEE and VVGAEE algorithms is studied in this section in terms of energy efficiency and computational efficiency.



(a) VVGAEE path and the fourth initial energy efficient path in the sea current of the first hour

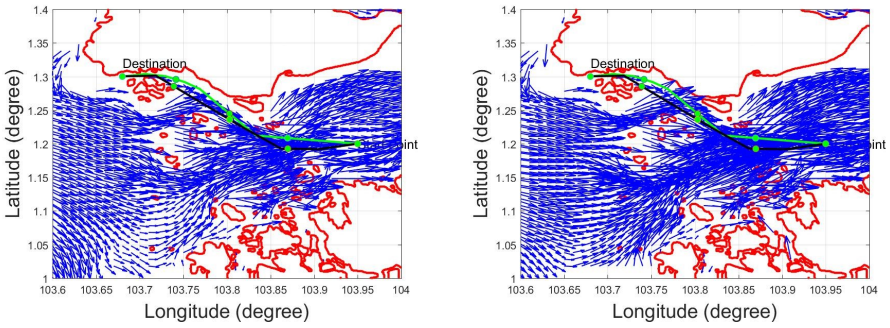
(b) VVGAEE path and the fourth initial energy efficient path in the sea current of the second hour

Fig. 15. Illustration of the VVGAEE path and the fourth initial energy efficient path: The green line represents the VVGAEE path, the black line represents the fourth initial energy efficient path and the green dot represents the one hour break point

To visualise the performance of each method, we demonstrate the performance using one USV mission first. A comprehensive performance analysis is carried out later with ten USV missions.

We use the same mission as section 4.2. By using the sea current data of the first hour, we can get the corresponding energy efficient path through applying the VVEE algorithm. The length of the path and the travel duration can be calculated. Assume the estimated travel duration is T hours (3.79 hours in this case). The T number of initial paths (4 in this case) will be used for generating 300 paths using crossover and mutation operators. Then 300 paths are evaluated and 120 best paths are selected as the parents of the next generation. These 120 best paths generate 120 new individuals by using the crossover operators and generate 60 mutational individuals by using the mutation operators. These 300 paths will be evaluated and selected again in the next iteration. The termination condition is the completion of 20 iterations of selection and evaluation, and the best individuals will be selected afterwards.

For comparison purpose, we visualise the VVGAEE path and the fourth initial energy efficient path with the sea states of the first and second hours in Fig. 15a and Fig. 15b. We can see that in the first two hours, when the



(a) VVGAEE path and the first initial energy efficient path in the sea current of the third hour

(b) VVGAEE path and the first initial energy efficient path in the sea current of the fourth hour

Fig. 16. Illustration of the VVGAEE path and the first initial energy efficient path: The green line is the VVGAEE path, the black line shows the first initial energy efficient path and the green dot highlights the one hour break point

direction of the sea current is westward, the VVGAEE path enables the USV to take advantage of the downstream instead of crossing the flow like the fourth initial path. It is obvious that the VVGAEE planned path will be more optimal in terms of energy efficiency than the fourth initial path in the case of sea states of the first two hours. As the first initial path and second initial path are similar, here we compare the VVGAEE path with the first initial path using the sea current data of the latter two hours, as shown in Fig. 16a and Fig. 16b. In this case, the direction of the sea current is eastward from west. The VVGAEE path directs the USV to travel towards north so that the sea current disturbance has less effect on the USV, instead of navigating upstream like the first initial path.

The energy consumption of the VVGAEE path and four initial path during each hour of this mission is given in Table 3. We can see that, in the first two hours, the VVGAEE path consumes considerably less energy than the third and the fourth paths, but consumes slightly more energy than the first and the second paths. In the latter two hours, the VVGAEE path consumes slightly more energy than the third and the fourth paths, but consumes less energy than the first and the second paths. Overall, the VVGAEE path requires the minimum

Table 3: The energy consumption of the VVGAEE path and four initial paths during each hour

	First hour	Second hour	Third hour	Fourth hour	Total
First path	0.3023	0.2347	0.7221	0.4690	1.7281
Second path	0.3129	0.1993	0.7153	0.4957	1.7231
Third path	0.3874	0.4789	0.5193	0.3440	1.7296
Fourth Path	0.3874	0.4789	0.5193	0.3440	1.7296
GA path	0.3312	0.2261	0.5848	0.4041	1.5462

energy in the whole mission duration and saves 10.27% energy than the second path, which is the most energy efficient one in the four paths produced by VVEE.

The VGAEE and VVEEPRE paths are also computed and illustrated in Fig. 17. The energy consumption for VVEEPRE and VGAEE are 1.6216 and 1.6741, which are 4.88% and 8.27% higher than the VVGAEE path respectively.

For a more comprehensive performance evaluation, we conducted 10 USV missions to compare the four algorithms. The energy consumption levels of the four algorithms are shown in Table 4. It can be seen that the VVEE, VVEEPRE and VGAEE algorithms consume more energy than the VVGAEE algorithm. Comparing VVEE and VVGAEE, VVEE needs 4.08% - 38.47% more energy than VVGAEE. Similarly, VVEEPRE requires 3.46% - 35.47% more energy than VVGAEE. The main cause of the performance difference is that VVEEPRE only searches for a local optimal path, while VVGAEE searches for a global optimal path. With VGAEE, it needs 0.84% - 41.65% more energy than VVGAEE, which demonstrates that using the initial paths generated in Voronoi-Visibility roadmap for the oncoming GA implementation has better energy efficiency than using the paths generated by the Voronoi roadmap (VGAEE). The desired mission speeds are set different for the missions, where mission 7 to mission 10 are deliberately set with incrementally increasing travel speeds but with the same start point to destination, and hence consume increasing energy levels. Overall, the VVGAEE algorithm achieves the best performance in terms of energy efficiency under different travel speeds.

To validate the practicality of the algorithms, the computational time of each algorithm is also studied. The Voronoi collision-free roadmap is first calculated

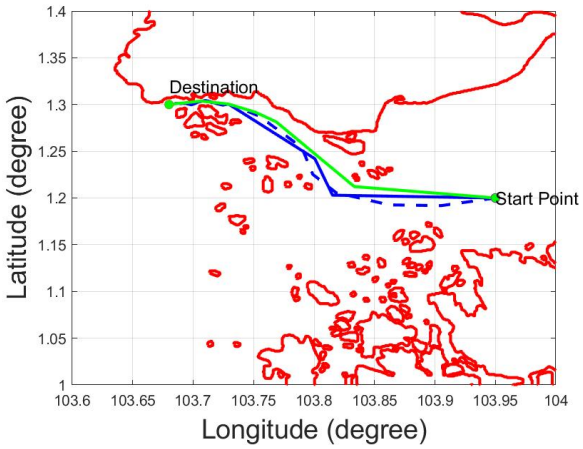


Fig. 17. Comparison of VVGAEE, VGAEE and VVEEPRE: green solid line represents the VVGAEE path, VGAEE is shown as blue solid line and blue dashed line is the VVEEPRE path

that takes around 30-40 seconds. The computational time of the oncoming algorithms are recorded, as shown in Table 5. Note that these three algorithms use the same Voronoi roadmap, which just needs to be calculated once and can be reloaded for future uses. It is found that the VVEEPRE algorithm needs much less computational time than VGAEE and VVGAEE. The reason is that VVEEPRE only requires executing the VVEE path planner for a few steps, while the other GA-based methods call the VVEE function for thousands of times. On the other hand, surprisingly, VVGAEE requires less computational time than VGAEE. Although VVGAEE calls the VVEE function to generate the initial paths instead of using the Voronoi paths directly, the initial VVEE paths are shorter than the Voronoi paths. This, in consequence, saves the ongoing computational time. And using the VVEE paths as the initial paths also improves the energy efficiency of the VGAEE algorithm, as shown in Table 4.

The relationship between travel time and computational time of VVGAEE is illustrated in Fig. 18. It is observed that the computational time of VVGAEE is linearly proportional to the travel time. In addition, comparing mission 4 (the longest travel time of 3.87 hrs) with mission 10 (the shortest travel time of 1.90

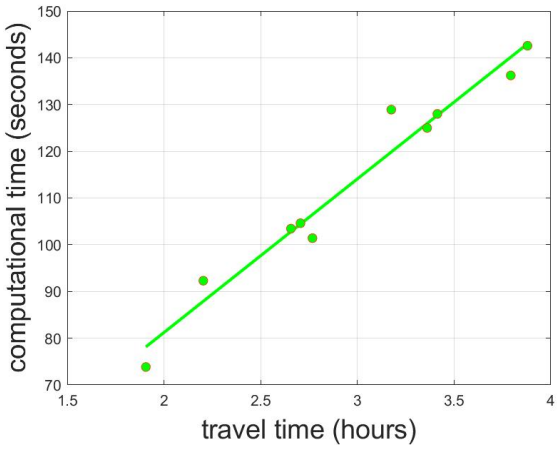


Fig. 18. Travel time and computational time of the VVGAEE algorithm in ten USV missions: each green dot represents the computational time against its corresponding travel time of one mission. The green line shows the linear regression fitting between travel time and computational time

hrs), we can also find the computational time of VVGAEE increases linearly and does not increase dramatically for searching high dimensional data like with A* and dynamic programming algorithms, demonstrating the scalability of the VVGAEE algorithm.

In total, the VVGAEE algorithm generates the most energy efficient paths, compared with the VVEE, VGAEE and VVEEPRE algorithms. Although the VVGAEE algorithm requires more computational time than the VVEEPRE algorithm, it is still practical considering that it uses two minutes to plan a 3-4 hours travel. Note that all the experiments were carried out on a 2.2GHz Intel Core i7-8750H processor with 16.0 GB. The program was implemented in Matlab R2019a. Using more powerful computing platform and optimising the code will reduce the computational time. As mentioned, the VVGAEE algorithm is considered scalable, due to the fact that the computational time of VVGAEE is linearly proportional to the travel time.

Table 4: Energy efficiency comparison of VVEE, VVEEPRE, VGAEE and VVGAEE in ten USV missions of Singapore Strait

No.	Start point	Destination	Speed (m/s)	VVEE energy cost	VVEEPRE energy cost	VGAEE energy cost	VVGAEE energy cost
1	(103.95, 1.20)	(103.75, 1.25)	2.5	1.026 (+14.13%)	1.026 (+14.13%)	0.943 (+4.89%)	0.899
2	(103.95, 1.20)	(103.70, 1.23)	2.5	1.609 (+38.47%)	1.262 (+8.61%)	1.646 (+41.65%)	1.162
3	(103.95, 1.15)	(103.70, 1.25)	2.5	1.786 (+27.21%)	1.532 (+9.12%)	1.478 (+5.27%)	1.404
4	(103.68, 1.30)	(103.95, 1.20)	2.5	1.654 (+12.36%)	1.523 (+3.46%)	1.569 (+6.59%)	1.472
5	(103.75, 1.25)	(103.95, 1.20)	2.5	1.430 (+4.08%)	1.461 (+6.33%)	1.388 (+1.02%)	1.374
6	(103.70, 1.25)	(103.95, 1.15)	2.5	1.854 (+12.16%)	1.827 (+10.53%)	1.706 (+3.21%)	1.653
7	(103.91, 1.27)	(103.70, 1.25)	2	0.946 (+11.82%)	1.067 (+26.12%)	0.863 (+2.01%)	0.846
8	(103.91, 1.27)	(103.70, 1.25)	2.5	1.431 (+12.41%)	1.421 (+11.63%)	1.300 (+2.12%)	1.273
9	(103.91, 1.27)	(103.70, 1.25)	3	2.000 (+29.70%)	2.089 (+35.47%)	1.555 (+0.84%)	1.542
10	(103.91, 1.27)	(103.70, 1.25)	3.5	2.572 (+20.47%)	2.667 (+ 24.92%)	2.530 (+18.50%)	2.135

Table 5: Computational time comparison of VVEEPRE, VGAEE and VVGAEE in ten USV missions of Singapore Strait

No.	Start point	Destination	Speed (m/s)	VVEEPRE computational time (seconds)	VGAEE computational time (seconds)	VVGAEE computational time (seconds)	VVGAEE distance (km)	VVGAEE travel time (hrs)
1	(103.95, 1.20)	(103.75, 1.25)	2.5	3.253	127.445	104.610	24.352	2.7058
2	(103.95, 1.20)	(103.70, 1.23)	2.5	4.449	152.288	128.902	28.577	3.1752
3	(103.95, 1.15)	(103.70, 1.25)	2.5	4.330	135.827	124.991	30.245	3.3606
4	(103.68, 1.30)	(103.95, 1.20)	2.5	6.103	170.562	142.571	34.915	3.8794
5	(103.75, 1.25)	(103.95, 1.20)	2.5	2.978	136.484	101.411	24.908	2.7676
6	(103.70, 1.25)	(103.95, 1.15)	2.5	5.327	159.493	136.217	34.137	3.7930
7	(103.91, 1.27)	(103.70, 1.25)	2	6.192	169.639	127.985	24.574	3.4131
8	(103.91, 1.27)	(103.70, 1.25)	2.5	4.559	130.730	103.442	23.907	2.6563
9	(103.91, 1.27)	(103.70, 1.25)	3	2.970	119.677	92.302	23.796	2.2033
10	(103.91, 1.27)	(103.70, 1.25)	3.5	2.874	113.068	73.841	24.018	1.9062

5. Conclusion and future work

In this work, we proposed a spatially-temporally energy efficient path planning algorithm, named VVGAEE, by integrating the Voronoi diagram, Visibility graph, Dijkstra's algorithm and GA algorithm. Voronoi diagram and Visibility graph are used to produce optimal roadmaps that are sparse and efficient for large-scale environment path planning. The use of the GA algorithm with various operators, including dividing, smoothing, and exchanging, is to further improve the quality of paths and adapt to the Voronoi-Visibility roadmap. For performance evaluation, the VVGAEE is evaluated against three other algorithms, namely VVEEPRE, VGAEE, and VVEE, where the first two are two new implementations inspired by other recent related work and the VVEE is one previous work, prividing a baseline for performance evaluation. A comprehensive study of performance is carried out with ten different USV missions. The VVGAEE algorithm has shown clear advantages in generating the most energy efficient path among all algorithms. Studies of computational time have also been performed in order to validate the practicability for large-scale real-world scenarios. The results have shown that the computational time of VVGAEE will increase only linearly with the increase of travel time, demonstrating the scalability of this algorithm. Using the historical sea current dataset and high-resolution coastline dataset of the real world also demonstrates the flexibility and practicability of the proposed algorithm.

In future, the main focus will be on the integration of the path planning algorithm and the previous works of the collision avoidance and the path following algorithm. Multiple energy resources, including wind energy (Ren et al., 2019), wave energy (Mutsuda et al., 2019), solar energy (García-Córdova and Guerrero-González, 2013) and diesel engine, may also be taken into account for long endurance operations. According to the COLREGS (International Regulations for Preventing Collisions at Sea), the USV may need to alter its course to avoid the encountering intruders in specific traffic situations. Path planning decision should be made accordingly to avoid collision risks. Multi-task alloca-

tion scenarios can be taken into account too. The sea depth data can also be integrated with the proposed algorithm so as to define the safe area dynamically in time varying environments.

References

Alvarez, A., Caiti, A., Onken, R., 2004. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. IEEE Journal of Oceanic Engineering 29 (2), 418–429.

Benavides, F., Tejera, G., Pedemonte, M., Casella, S., 2011. Real path planning based on genetic algorithm and voronoi diagrams. In: Robotics Symposium, 2011 IEEE IX Latin American and IEEE Colombian Conference on Automatic Control and Industry Applications (LARC). IEEE, pp. 1–6.

Bryson, A., Ho, Y., 1975. Applied optimal control. Hemisphere, New York, 458.

Carroll, K. P., McClaran, S. R., Nelson, E. L., Barnett, D. M., Friesen, D. K., William, G., 1992. Auv path planning: an a* approach to path planning with consideration of variable vehicle speeds and multiple, overlapping, time-dependent exclusion zones. In: Autonomous Underwater Vehicle Technology, 1992. AUV'92., Proceedings of the 1992 Symposium on. IEEE, pp. 79–84.

Chen, J., Zhu, H., Zhang, L., Sun, Y., 2018. Research on fuzzy control of path tracking for underwater vehicle based on genetic algorithm optimization. Ocean Engineering 156, 217–223.

Eichhorn, M., 2015. Optimal routing strategies for autonomous underwater vehicles in time-varying environment. Robotics and Autonomous Systems 67, 33–43.

Fu, X., Lei, L., Yang, G., Li, B., 2018. Multi-objective shape optimization of autonomous underwater glider based on fast elitist non-dominated sorting genetic algorithm. Ocean Engineering 157, 339–349.

Garau, B., Alvarez, A., Oliver, G., 2005. Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an a* approach. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. IEEE, pp. 194–198.

Garau, B., Bonet, M., Alvarez, A., Ruiz, S., Pascual, A., 2014. Path planning for autonomous underwater vehicles in realistic oceanic current fields: Application to gliders in the western mediterranean sea. Journal of Maritime Research 6 (2), 5–22.

García-Córdova, F., Guerrero-González, A., 2013. Intelligent navigation for a solar powered unmanned underwater vehicle. International Journal of Advanced Robotic Systems 10 (4), 185.

Goldberg, D. E., Holland, J. H., 1988. Genetic algorithms and machine learning. Machine learning 3 (2), 95–99.

Khalili-Damghani, K., Abtahi, A.-R., Tavana, M., 2013. A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. Reliability Engineering & System Safety 111, 58–75.

Kim, H., Kim, S.-H., Jeon, M., Kim, J., Song, S., Paik, K.-J., 2017. A study on path optimization method of an unmanned surface vehicle under environmental loads using genetic algorithm. Ocean Engineering 142, 616–624.

Kularatne, D., Bhattacharya, S., Hsieh, M. A., 2016. Time and energy optimal path planning in general flows. In: Robotics: Science and Systems.

Le, A., Prabakaran, V., Sivanantham, V., Mohan, R., 2018. Modified a-star algorithm for efficient coverage path planning in tetris inspired self-reconfigurable robot with integrated laser sensor. Sensors 18 (8), 2585.

Lee, T., Kim, H., Chung, H., Bang, Y., Myung, H., 2015. Energy efficient path planning for a marine surface vehicle considering heading angle. Ocean Engineering 107, 118–131.

Li, P., Duan, H., 2012. Path planning of unmanned aerial vehicle based on improved gravitational search algorithm. Science China Technological Sciences 55 (10), 2712–2719.

Lolla, T., Haley, P. J., Lermusiaux, P. F., 2014. Time-optimal path planning in dynamic flows using level set equations: realistic applications. Ocean Dynamics 64 (10), 1399–1417.

Lolla, T., Haley Jr, P., Lermusiaux, P., 2015. Path planning in multi-scale ocean flows: Coordination and dynamic obstacles. Ocean Modelling 94, 46–66.

Lolla, T., Ueckermann, M., Yiğit, K., Haley, P. J., Lermusiaux, P. F., 2012. Path planning in time dependent flow fields using level set methods. In: Robotics and Automation (ICRA), 2012 IEEE International Conference on. IEEE, pp. 166–173.

Lv, C., Yu, H., Chi, J., Xu, T., Zang, H., lue Jiang, H., Zhang, Z., 2019. A hybrid coordination controller for speed and heading control of underactuated unmanned surface vehicles system. Ocean Engineering 176, 222–230.

Ma, Y., Hu, M., Yan, X., 2018. Multi-objective path planning for unmanned surface vehicle with currents effects. ISA transactions 75, 137–156.

Mutsuda, H., Tanaka, Y., Doi, Y., Moriyama, Y., 2019. Application of a flexible device coating with piezoelectric paint for harvesting wave energy. Ocean Engineering 172, 170–182.

Niu, H., Lu, Y., Savvaris, A., Tsourdos, A., 2018. An energy-efficient path planning algorithm for unmanned surface vehicles. Ocean Engineering 161, 308–321.

Palomeras, N., Vallicrosa, G., Mallios, A., Bosch, J., Vidal, E., Hurtos, N., Carreras, M., Ridao, P., 2018. Auv homing and docking for remote operations. Ocean Engineering 154, 106–120.

Pehlivanoglu, Y. V., 2012. A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous uav. Aerospace Science and Technology 16 (1), 47–55.

Ren, Z., Skjetne, R., Gao, Z., 2019. A crane overload protection controller for blade lifting operation based on model predictive control. Energies 12 (1), 50.

Shiltagh, N. A., Jalal, L. D., 2013. Path planning of intelligent mobile robot using modified genetic algorithm. International Journal of Soft Computing and Engineering (IJSCE) 3 (2), 31–36.

Singh, Y., Sharma, S., Sutton, R., Hatton, D., Khan, A., 2018. A constrained a* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents. Ocean Engineering 169, 187–201.

Song, C. H., 2014. Global path planning method for usv system based on improved ant colony algorithm. In: Applied Mechanics and Materials. Vol. 568. Trans Tech Publ, pp. 785–788.

Song, R., Liu, Y., Bucknall, R., 2017. A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment. Ocean Engineering 129, 301–317.

Song, R., Liu, Y., Bucknall, R., 2019. Smoothed a* algorithm for practical unmanned surface vehicle path planning. Applied Ocean Research 83, 9–20.

Tang, B., Zhu, Z., Luo, J., 2016. Hybridizing particle swarm optimization and differential evolution for the mobile robot global path planning. International Journal of Advanced Robotic Systems 13 (3), 86.

TideTech, 2017. Welcome to tidetech. [Online; accessed 19-June-2017].
URL `https://www.tidetech.org/data/`

Xia, Y., Xu, K., Li, Y., Xu, G., Xiang, X., 2019. Improved line-of-sight trajectory tracking control of under-actuated auv subjects to ocean currents and input saturation. Ocean Engineering 174, 14–30.

Yang, P., Tang, K., Lozano, J. A., Cao, X., 2015. Path planning for single unmanned aerial vehicle by separately evolving waypoints. IEEE Transactions on Robotics 31 (5), 1130–1146.

Yilmaz, N. K., Evangelinos, C., Lermusiaux, P. F., Patrikalakis, N. M., 2008. Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming. IEEE Journal of Oceanic Engineering 33 (4), 522–537.

Zadeh, S. M., Powers, D. M., Sammut, K., Yazdani, A., 2016. Differential evolution for efficient auv path planning in time variant uncertain underwater environment. arXiv preprint arXiv:1604.02523.

Zeng, Z., Lian, L., Sammut, K., He, F., Tang, Y., Lammas, A., 2015. A survey on path planning for persistent autonomy of autonomous underwater vehicles. Ocean Engineering 110, 303–313.

Zhang, Y., Zhang, L., Zhang, X., 2008. Mobile robot path planning base on the hybrid genetic algorithm in unknown environment. In: Intelligent Systems Design and Applications, 2008. ISDA'08. Eighth International Conference on. Vol. 2. IEEE, pp. 661–665.